

## GEOMETRIA COMPUTACIONAL

PEDRO GIGECK FREIRE

10737136

LISTA 8

**5** DESCREVA um ALGORITMO que constrói o fecho convexo de um polígono Ymonótono em tempo linear.

IDEIA: Começar com um ponto que faz parte do fecho  
(por exemplo o mais baixo, mais à esquerda)

E ir colocando os pontos montando o fecho

Pra cada ponto colocado, retirar os que ficaram "pra de lado"

(Parecido com o algoritmo incremental)

Ponto Baixo ( $P, n$ )

- 1 | baixo  $\leftarrow 1$
- 2 | para  $i \leftarrow 2$  até  $n$  faça
- 3 |     se  $P[i].y < P[baixo].y$  ou  $(P[i].y = P[baixo].y \text{ e } P[i].x < P[baixo].x)$
- 4 |         então baixo  $\leftarrow i$
- 5 | devolva baixo

Fecho Convexo Monotônico ( $P, n$ )

- 1 | baixo  $\leftarrow$  Ponto Mais Baixo ( $P, n$ )
- 2 |  $H[1] \leftarrow P[baixo]$
- 3 |  $h \leftarrow 1$
- 4 | atual  $\leftarrow$  baixo + 1
- 5 | se atual =  $n+1$
- 6 |     então atual  $\leftarrow 1$

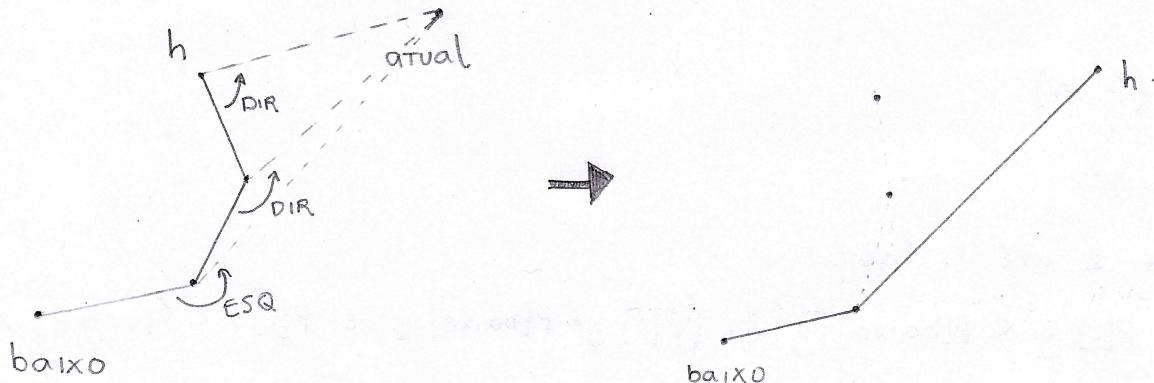
```

7   enquanto atual ≠ baixo faça
8       |   enquanto  $h > 1$  e  $\text{Dir}(H[h-1], H[h], P[\text{atual}])$  faça
9           |       |    $h \leftarrow h - 1$ 
10          |       |    $h \leftarrow h + 1$ 
11          |       |    $H[h] \leftarrow P[\text{atual}]$ 
12         |       |    $\text{atual} \leftarrow \text{atual} + 1$ 
13         |       se  $\text{atual} = n + 1$ 
14             |           então  $\text{atual} \leftarrow 1$ 
15   Devolva  $(H, h)$ 

```

---

Simulando alguma iteração



Como temos  $n-1$  iterações, em cada iteração RETIRAMOS Alguns pontos do fecho, mas cada ponto é retirado no máximo 1 vez, então o laço mais interno será executado no máximo, no total,  $n$  vezes. Então consumimos tempo linear.

## 8 Diâmetro e Largura

(a) DEFINA O DIÂMETRO DE UM CONJUNTO DE PONTOS  $\{P_1, \dots, P_n\}$  COMO A DISTÂNCIA MÁXIMA ENTRE QUAISQUER DOIS PONTOS DO CONJUNTO

$$\text{DIÂMETRO } (P_1, \dots, P_n) = \max_{i,j} |P_i - P_j|.$$

Prove que o diâmetro de um conjunto é atingido por dois vértices do fecho convexo do conjunto.

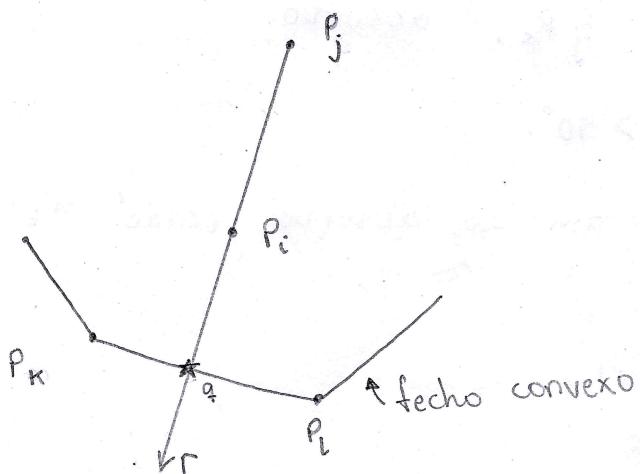
Seja  $P_i, P_j$  o par de pontos mais distantes da coleção.

Suponha, por absurdo, que  $P_i$  não faça parte do fecho convexo.

Como  $P_i$  está no interior do fecho convexo, então qualquer

reta que passa por  $P_i$  intersecta com o fecho.

Vamos construir uma reta  $r$  na direção do segmento  $P_j P_i$ :



Seja  $q$  o ponto de interseção entre  $r$  e o fecho convexo, no sentido de  $P_j \rightarrow P_i$ , ou seja, é o ponto mais próximo ao lado  $P_j P_i$ .

Se  $|q - P_j| > |P_i - P_j|$ , é absurdo.

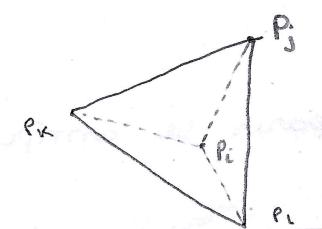
Se  $q \in \{P_1, \dots, P_n\}$ , então  $|q - P_j| > |P_i - P_j|$ , absurdo.

Se  $q \notin \{P_1, \dots, P_n\}$ , então ele está em uma aresta do fecho, seja

$P_k, P_l$  os extremos dessa aresta.

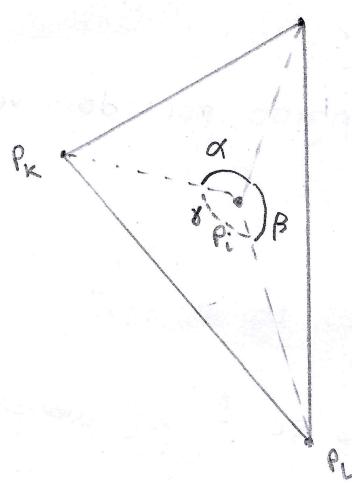
Vamos montar um triângulo  $P_k P_i P_j$  com TRÊS TRIÂNGULOS DENTRO

(Cada aresta juntando com o  $P_i$  como vértice).



Seja  $\alpha$  o ângulo em  $P_k P_i P_j$  e

$\beta$  o ângulo em  $P_l P_i P_j$ .



Como o complementar de  $\alpha + \beta$ ,  $\gamma$  faz parte de um TRIÂNGULO, então

$$\gamma < 180^\circ \Rightarrow 360^\circ - \alpha - \beta < 180^\circ \Rightarrow$$

$$\underline{\alpha + \beta > 180^\circ}$$

Se  $\alpha = \beta = 90^\circ$ , então  $P_i, P_j, P_k$  é um TRIÂNGULO retângulo com  $P_jP_k$  de hipotenusa, então  $|P_j - P_k| > |P_j - P_i|$ , ABSURDO.

Se  $\alpha > 90^\circ$ , então  $P_i, P_j, P_k$  é um TRIÂNGULO com o maior ângulo em  $P_i$ , então maior lado em  $P_jP_k$ , ABSURDO.

O mesmo raciocínio se  $\beta > 90^\circ$ .

Portanto, TODOS os casos levaram ao ABSURDO, então  $P_i$  pertence ao fecho convexo.

Análogo para o  $P_j$ .

$P_i$  e  $P_j$  tem que pertencer ao fecho convexo.

(b) Uma reta de suporte de um conjunto é uma reta que toca o fecho convexo deste conjunto e deixa TODOS os pontos de um mesmo lado, ou em uma da reta. [Prove que o DIÂMETRO de um conjunto é o mesmo que a distância máxima entre retas de suporte para o conjunto.]

No exercício anterior, provamos que o diâmetro ocorre entre dois pontos do fecho convexo.

E esses pontos pertencem a retas de suporte, então temos que provar que essas retas são as mais distantes dentre todas.

Recapitulando, a distância entre duas retas é:

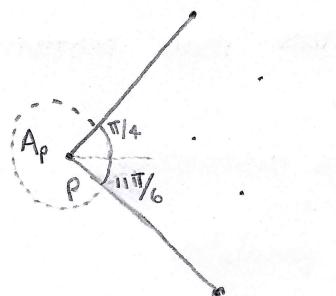
$$|r_1 - r_2| = \begin{cases} 0, & \text{se } r_1 \text{ e } r_2 \text{ são concorrentes} \\ \min(|P_i - P_j|) : P_i \in r_1 \text{ e } P_j \in r_2, & \text{se } r_1 \text{ e } r_2 \text{ são paralelas} \end{cases}$$

Então primeiro vamos provar que os pontos do diâmetro aceitam retas de suporte paralelas.

Sejam  $p$  e  $q$  os pontos mais distantes da coleção.

Seja  $A_p$  o conjunto de todos os ângulos disponíveis para retas de suporte apoiadas em  $p$ .

Por exemplo

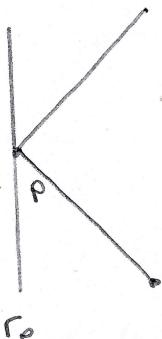


$$A_p = [\pi/4, 11\pi/6]$$

Vamos traçar uma reta 'tangente' a  $p$ , com 'inclinação média em  $A_p$ '

$$\text{A inclinação da reta será } \theta_p = \frac{\min(A_p) + \max(A_p)}{2} + \frac{\pi}{2}$$

Então no nosso exemplo



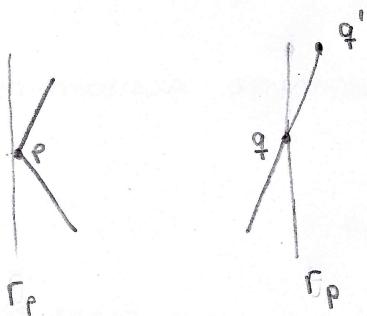
$$\theta_p = \frac{\pi/4 + 11\pi/6}{2} + \frac{\pi}{2} = \frac{37\pi}{24} (\approx \frac{3\pi}{2})$$

→ Reta 'tangente' a  $p$  com inclinação  $\theta_p$

Agora temos que provar que existe uma reta com essa inclinação em  $q$ .

Suponha, por absurdo, que  $q$  não aceite uma reta de suporte com inclinação  $\theta_p$ .

Então existe um outro ponto  $q'$  tal que a reta  $q, q'$  do fecho convexo intersecta a reta  $r_q$  com inclinação  $\theta_p$ .



Mas então, o triângulo  $p-q-q'$  ficaria com o lado maior  $p-q'$ , o que é absurdo.

Então os pontos  $p, q$  tem duas retas de suporte paralelas  $r_p$  e  $r_q$ .

Temos que provar que esse é o par de retas mais distante.

Seja  $s_1, s_2$  um par de retas de suporte mais distantes, como  $|s_1 - s_2| > |r_p - r_q| > 0$ , pois  $r_p$  e  $r_q$  são paralelas.

Então  $s_1$  e  $s_2$  são paralelas.

Seja  $p \in s_1$  e  $q \in s_2$  os pontos do fecho convexo em  $s_1$  e  $s_2$ ,

Temos que

$$|p - q'| \geq |s_1 - s_2| = \min_{\substack{P_i \in s_1, \\ P_j \in s_2}} |P_i - P_j|$$

Portanto

$$|p - q'| \geq |s_1 - s_2| \geq |r_p - r_q| = |p - q|$$

Como  $p$  e  $q$  são os pontos mais distantes, então  $|p - q'| = |p - q|$

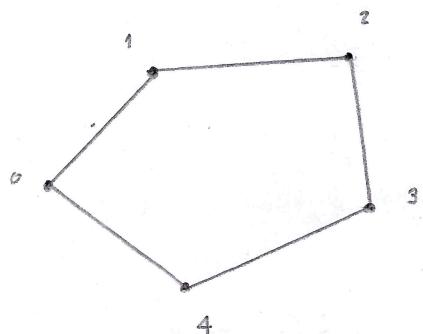
Então o Diâmetro  $|p - q|$  é igual a maior distância entre retas de suporte

(c) Dois pontos  $a$  e  $b$  são antípodas se eles estão em retas de suporte paralelas (distintas). Existem duas retas de suporte paralelas (distintas) uma passando por  $a$  e a outra por  $b$ .

Projete um algoritmo para listar TODOS os pares de antípodas de um conjunto de pontos no plano.

IDEIA: VAMOS RODANDO pelo fecho convexo Descobrindo os pares de antípodas, RODANDO em paralelo nos opostos do fecho.

Por exemplo:

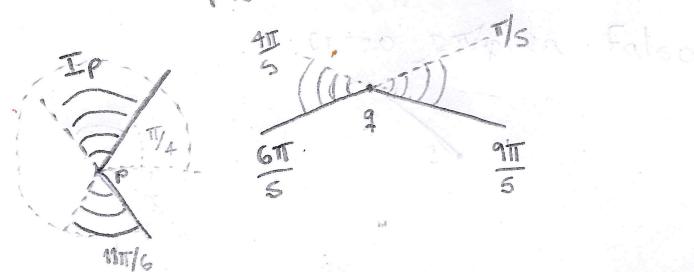


- Comegamos com o ponto 0 e encontramos o primeiro par 0-2
- Vamos avançando adicionando os pares com 0
- Pararmos para o ponto 1 e encontrarmos o primeiro par, mas buscando a partir de 2.
- E assim sucessivamente

Primeiro, temos que fazer uma função para testar se um par é antípoda.

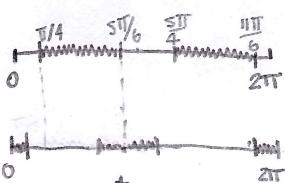
Vamos ver o intervalo de retas de suporte que cada ponto aceita e depois verificar se há uma intersecção entre eles.

Por exemplo



$$I_p = [\pi/4, \frac{5\pi}{6}] \cup [\frac{5\pi}{4}, \frac{11\pi}{6}]$$

$$I_q = [\frac{4\pi}{5}, \frac{6\pi}{5}] \cup [\frac{9\pi}{5}, 2\pi] \cup [0, \frac{\pi}{5}]$$



(Desenho super fora de escala.)

Como  $I_p \cap I_q \neq \emptyset$

Então os pontos são antípodas

(Aceitam retas paralelas com inclinação  $I_p \cap I_q$ )

## Antipoda ( $H, i, j, h$ )

```
1 |   p ← H[i], q ← H[j]
2 |   Ip ← intervalo Suporte(H, i, h)
3 |   Iq ← intervalo Suporte(H, j, h)
4 |   para i em Ip faça
5 |       para j em Iq faça
6 |           se j[0] < i[0] < j[1] ou          // i ∩ j ≠ ∅
7 |               j[0] < i[1] < j[1]
8 |               então devolva Verdadeiro
9 |   devolva falso
```

Como os intervalos de retas de suporte  $I_p$  e  $I_q$  são uniões de, no máximo 3 intervalos, então esse teste consome tempo constante.

## intervalo Suporte ( $H, i, h$ )

```
1 |   prev ← i-1
2 |   prox ← i+1
3 |   se prox = 0
4 |       então prox = h
5 |   se prox = h+1
6 |       então prox = 1
7 |   Ip = {[Angulo(p, prev), Angulo(p, prox) - π], [Angulo(p, prox), Angulo(p, prev) - π]}
8 |   ▷ Talvez  $I_p$  precise de alguma formatação especial, vou supor que não
   |   devolva  $I_p$ 
```

Agora vamos ao algoritmo de fato

## Antipodas ( $P, n$ )

```
1 |   ( $H, h$ )  $\leftarrow$  FechoConvexo ( $P, n$ )
2 |   candidatoMin  $\leftarrow 0$     p  $\leftarrow 0$ 
3 |   resp  $\leftarrow \emptyset$ 
4 |   enquanto candidatoMin  $< h$  faça
5 |       enquanto não Antipoda ( $p, candidatoMin$ ) faça
6 |           candidatoMin  $\leftarrow$  candidatoMin + 1
7 |           q  $\leftarrow$  candidatoMin
8 |           enquanto Antipoda ( $p, q$ )
9 |               resp  $\leftarrow$  resp  $\cup \{p, q\}$ 
10 |              q  $\leftarrow$  q + 1
11 |          p  $\leftarrow$  p + 1
12 |   devolva resp
```

Embora o algoritmo pareça quadrático, podemos ver que o candidatoMin sempre incrementa pelo menos uma vez a cada iterações do laço externo. O primeiro laço interno executa, no total,  $n$  vezes, pois ele que incrementa nosso candidatoMin.

O segundo laço interno executa, no total, o número de pares de antipodas na coleção, que é linear no número de pontos no fecho.

Então o algoritmo consome tempo linear (Além do tempo de obter o fecho convexo, conforme discutido no fórum).

(d) Defina a largura como a distância mínima entre duas retas de suporte paralelas (distintas). Projete um algoritmo para calcular a largura de um conjunto de pontos no plano.

Esse algoritmo fica quase trivial se usarmos o algoritmo do exercício anterior para obter TODOS os pares de antípodas.

langura ( $P, n$ )

- 1 |     $A \leftarrow \text{Antípodas}(P, n)$
- 2 |     $l \leftarrow +\infty$
- 3 |    para  $p, q$  em  $A$  faça
- 4 |     |     $l \leftarrow \min(l, |p-q|)$
- 5 |    devolva  $l$